USENIX Security '25 Artifact Appendix: Robustifying ML-powered Network Classifiers with PANTS

Minhao Jin
Princeton University

Maria Apostolaki
Princeton University

A Artifact Appendix

A.1 Abstract

PANTS is a practical framework for assessing and enhancing the robustness of ML-based network traffic classification (MNC) using adversarial, realizable, and semantics-preserving inputs. This artifact includes all the necessary code and scripts for PANTS, along with supplementary resources such as datasets and well-trained models including both vanilla and robustified ones. Additionally, it provides a comprehensive roadmap for effectively utilizing and evaluating PANTS.

A.2 Description & Requirements

A.2.1 Security, privacy, and ethical concerns

The execution of artifacts will not raise any risk on security, privacy, or ethical concerns. All the datasets required to evaluate artifacts are public.

A.2.2 How to access

To better evaluate PANTS for its functionality and reproducibility, we provide an instance in CloudLab that has already installed all the required dependencies, data, code, and scripts such that the user can test PANTS smoothly. For any reviewer who would like to use our provided instance, please send your public key via comments so that we can add you to the access list for the provided machine. After we add the public key, reviewers can easily access via

```
$ ssh minhaoj@nfs.usenix-artifact.netsyn.emulab.net
```

After logging into the instance, the PANTS codebase is in the directory `/nfs/PANTS`. Please note that the CloudLab instance will be terminated after the artifact evaluation period. However, the artifact codebase as well as the corresponding data and models will be open to the public permanently.

If reviewers prefer to evaluate locally, the latest version of the source code is provided in the https://github.com/jinminhao/PANTS. For artifact evaluation purposes, reviewers can refer to the latest commit. Reviewers need to download some secondary artifacts, such as datasets and trained models, from an external Google Drive link. Details of downloading these secondary artifacts will be introduced in §A.3. In the provided CloudLab instance, all these files are already downloaded.

We release the finalized version in both https://github.com/jinminhao/PANTS and https://zenodo.org/records/14756845.

A.2.3 Hardware dependencies

If the reviewer prefers to test locally, we recommend evaluating PANTS using the machine that has similar or identical performance as c220g5 or d430 provided by CloudLab. For example, the c220g5 machine has two Intel Xeon Silver 4114 10-core CPUs at 2.20 GHz with 192GB ECC DDR4-2666 Memory.

Please refer to the CloudLab hardware page (https://docs.cloudlab.us/hardware.html) for more details on the hardware specification.

A.2.4 Software dependencies

To execute the artifacts, the testbed must be running on a Linux operating system. Additionally, Conda must be installed on the testbed. All required software dependencies are listed in the environment specification files (requirements-app-vpn.txt and requirements-vca.txt), which is provided within the codebase along with detailed instructions on how to execute these scripts. We provide a detailed guide on installation in §A.3.1.

A.2.5 Benchmarks

We evaluate PANTS with three different applications, which would require three datasets. All the datasets and the corresponding well-trained models (including vanilla and robustified ones) have been provided via a shared google drive folder. Please refer to §A.3.1 for details on downloading them.

A.3 Set-up

A.3.1 Installation

Please note that installation is only required if reviewers are evaluating PANTS locally. For any evaluation using the pro-

vided machine, the environment has already been configured and it is not necessary to re-install.

Installation of PANTS basically includes three steps:

- Step 1: Make sure the conda is correctly installed. Please refer to the conda page for more details.
- Step 2: Download the asset from this Google Drive link and decompress the `asset.tar.gz`. After decompression, you should see an 'asset' directory which includes data and models.
- Step 3: Run the following code to install all the required dependencies. Basically, running the artifacts require two different virtual environments.

```
$ cd PANTS/
$ conda create -n py39-app-vpn python=3.9
$ conda activate py39-app-vpn
$ pip3 install -r requirements-app-vpn.txt
$ conda create -n py39-vca python=3.9
$ conda activate py39-vca
$ pip3 install -r requirements-vca.txt
```

A.3.2 Basic Test

After all the dependencies are installed, reviewers can easily test PANTS by running

```
$ cd PANTS/scripts/
$ bash test-env.sh
```

To briefly introduce, this script uses PANTS to attack two models under two different applications (APP and QOE). If the environment is successfully configured, it should return with outputs like the following:

```
$ bash test-env.sh

...
Summary: ASR: 1.0, speed: 2.0500868564283423
...
Summary: ASR: 1.0, speed: 0.5714318749236006
```

Please note that the script will automatically activate the required environment based on the target application, i.e., when PANTS is running against APP(QOE), it will switch to the environment py39-app-vpn(py39-vca) automatically.

A.4 Evaluation workflow

A.4.1 Major Claims

(C1): PANTS is 70% more likely to find adversarial samples compared to Amoeba in median and 2x more likely compared to BAP. This is reflected in Fig. 6 in Finding 1. It describes the ASR when using PANTS and baselines to find adversarial samples, and it shows that PANTS can find adversarial samples with a high success rate.

(C2): Iterative augmentation with adversarial, realizable, and semantics-preserving samples improves the robustness of an MNC without hurting its accuracy. It corresponds to Fig. 7 and Fig. 8 in Finding 2. Fig. 7 describes the accuracy and ASR for models robustified by different approaches. It shows that PANTS can robustify models without sacrificing model accuracy. Fig. 8 plots the improvement of ASR during iterative robustification using PANTS.

(C3): PANTS improves the robustness of the tested MNCs 142% more than Amoeba and 40X more than NetShare. This claim is in the Finding 3. The corresponding figures are Fig. 9 and Fig. 10. Fig. 9 shows the accuracy and ASR when leveraging NetShare for robustfication. Fig. 10 describes the ASR for vanilla, PANTS-robustified and Amoeba-robustified models.

(C4): PANTS can improve the robustness of MNCs even against threat models outside those used during adversarial training. Fig. 11 in Finding 4 shows the ASR of two APP implementations for various threat models.

(C5): PANTS is practical and efficient to use, generating $\sim$1.7 samples/sec, which is 8X faster than Amoeba. Fig. 12 in Finding 5 plots the generation speed for PANTS and Amoeba.

(C6): Transferability provides an opportunity to robustify an MNC without white-box access. This is mainly discussed in Finding 6. Fig. 13 presents the result of transferability.

A.4.2 Experiments

Strictly running all the experiments would require weeks on a 5-machine cluster. Therefore, we only report PANTS' result on a **subset** of data for each experiment, which still needs $\sim$46 compute-hours when running on our provided instance. As baselines need to train on a full set of data, which is very time-consuming, evaluation of them is not included in this artifact. Reviewers can refer to the results in the paper for their performance.

(E-all): [1 human-minute + 46 computer-hours + 20GB disk]:

How to: Reviewers can run all the experiments using a single script. The script automatically runs all the provided testing scripts from (E1) to (E5) to get the results and generate the corresponding figures.

In short, it is easy to run (E-all) by

```
$ cd PANTS/scripts
$ bash test-all.sh
```

Results: All the results are presented as figures in the directory `PANTS/figures`. For each generated figure, please refer to the following (E_i) for a detailed explanation.

(E1): [1 human-minute + 25 compute-hour + 20GB disk]: (E1) generally includes 2 parts. First, it evaluates

PANTS' effectiveness in finding adversarial samples. This is mainly done by leveraging PANTS to find adversarial samples against vanilla models under different applications. Then, we evaluate the ASR to find adversarial samples against models robustified by different methods such as PANTS-robustified, Amoeba-robustified and NetShare-robustified ones. We show that PANTS can robustify MNCs more than Amoeba and NetShare cannot robustify the MNCs. (E1) demonstrates our major claim of (C1), (C3) and (C5).

How to: Reviewers can run the following two scripts sequentially to evaluate (E1)

```
$ cd PANTS/scripts
$ bash test-asr.sh
$ bash test-netshare.sh
```

Results: The majority of results are directly plotted as figures in `PANTS/figures`. After the two scripts finish, there should be multiple figures generated to validate (C1), (C3) and C(5). The first batch are

- `app_end_host_vanilla.pdf`
- `app_in_path_vanilla.pdf`
- `vpn_end_host_vanilla.pdf`
- `vpn_in_path_vanilla.pdf`
- `vca_end_host_vanilla.pdf`
- `vca_in_path_vanilla.pdf`

They show the ASR when using PANTS to against vanilla models under different applications and threat models. The corresponding figure in the paper is Fig. 6. The second batch shows the ASR on using PANTS against PANTS-robustified, Amoeba-robustified, and NetShare-robustified models. They are

- `app_end_host_robustified.pdf`
- `app_in_path_robustified.pdf`
- `vpn_end_host_robustified.pdf`
- `vpn_in_path_robustified.pdf`
- `vca_end_host_robustified.pdf`
- `vca_in_path_robustified.pdf`
- `netshare_mlp.pdf`
- `netshare_rf.pdf`
- `netshare_tf.pdf`
- `netshare_cnn.pdf`

Their corresponding figures and table are Fig. 9, Fig. 10 and Table 2 in the paper.

Finally, the generation speed for Fig. 12 is logged in `PANTS/logs/{app,vpn,vca}/..._vanilla/results.txt`.

(E2): [1 human-minute + 9 compute-hour + 20GB disk]: (E2) also consists of two parts. First, it evaluates the model accuracy and robustness for multiple robustification approaches. Then, it shows the improvement of ASR when using PANTS' generated samples for iteratively robustifying the models. (E2) demonstrates our major claim of (C2).

How to: Users are required to run two scripts sequentially

```
$ cd PANTS/scripts
$ bash test-adv-train.sh
$ bash test-robustification.sh
```

Results: The script plots figures

- `adv_train.pdf`
- `robustfication_end-host.pdf`

Please refer to Fig. 7 and Fig. 8 in the original paper for validation.

(E3): [1 human-minute + 11 compute-hour + 20GB disk]: (E3) evaluates the PANTS-robustified models against threat models outside those used during robustification. It mainly validates the major claim C4.

How to: Users should run one script

```
$ cd PANTS/scripts
$ bash test-more-threat.sh
```

Results: The script plots the figure, `threat.pdf`. It corresponds to Fig. 11.

(E4): [1 human-minute + 1 compute-hour + 20GB disk]: (E4) mainly evaluates the transferability for the samples generated by PANTS. It corresponds to the major claim (C6).

How to: Users are required to run one script

```
$ cd PANTS/scripts
$ bash test-transferability.sh
```

Results: The script plots the figure, `transferability.pdf`. Its corresponding figure is Fig. 13.

A.5 Version

Based on the LaTeX template for Artifact Evaluation V20231005. Submission, reviewing and badging methodology followed for the evaluation of this artifact can be found at https://secartifacts.github.io/usenixsec2025/.